A Ptolemy II Implementation of the TRANSCEND system for Model Based Fault Detection and Isolation in Continuous Dynamic Systems

Eric-J. Manders
Vanderbilt University
Department of Electrical Engineering and Computer Science

Ptolemy Mini-Conference, Berkeley, California, 22-23 March 2001

Outline

- Introduction: What, Why, and So What
- TRANSCEND:
  Framework for Model Based Fault Detection and Isolation
- Implementation in Ptolemy II
- Example: Multi-tank Fluid System
- Summary and plans for further work

Introduction

- **What**: Computational framework for Model Based FDI
- **Why**: Complex computational issues
  - Multiple concurrent tasks:
    signal processing, diagnosis algorithms, visualization
  - Online FDI subject to real-time requirements
- **So What**: Long term goal: integrate FDI as embedded system application for "smart physical systems"

Model Based Fault Detection and Isolation

- FDI with functional models
  - Exploit analytical redundancy in a system model
  - Require estimate of nominal behavior
  - Deviation from nominal behavior triggers FDI
- Many Approaches

	CT	D(T)
Quantitative	State/Parameter Estimation	Constraint analysis (AI)
Qualitative	TRANSCEND	Discrete Event Approaches

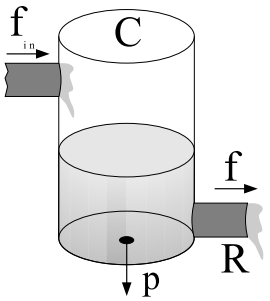
Model Based Fault Detection and Isolation (2)

- Arguments for the qualitative approach
  - Model accuracy (structural + parameter estimation)
  - Computational problems in numerical solution methods
    (e.g. convergence)
- Problems with traditional AI modeling approaches: under-constrained models lead to many spurious hypotheses

TRANSCEND: transient based diagnosis

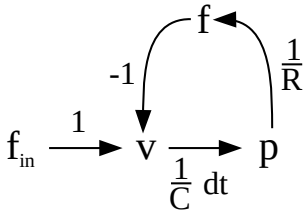
- Abrupt faults in continuous dynamic systems
  - Abrupt faults introduce Transients
  - Fault isolation is based on transient analysis
- Description of transient dynamics:
  qualitative magnitude and derivative values (-,0,+)
- Incremental analysis of transient behavior
- Topological Models based on physical principles: graph representation provides direct parameter to measurement relations

TRANSCEND: modeling for FDI

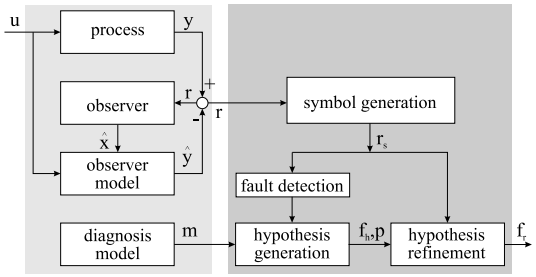


$$f = \frac{p}{R}$$

$$\dot{p} = \frac{1}{C}(f_{in} - f)$$

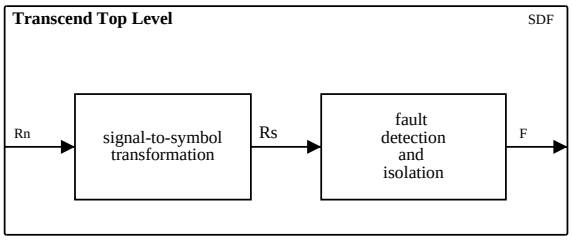


TRANSCEND: system architecture



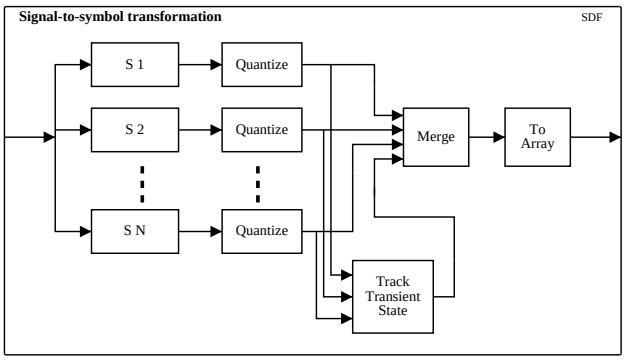
- Key properties
  - Robust methods for signal to symbol transformation
  - Hypothesis generation: graph algorithms
  - Hypothesis refinement: qualitative fault observers

Top level actor



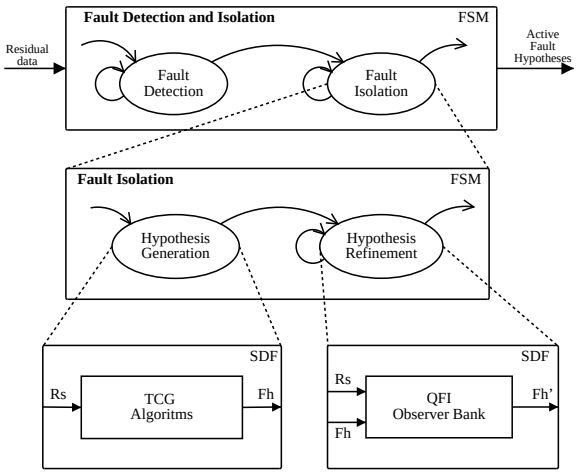
- Input: read residual data from file in off-line use
- Output: display active hypotheses as formatted text

Signal-to-Symbol transformation



- Derivative estimates computed with FIR filters
- Transient state controls hypothesis refinement

Qualitative Fault Detection and Isolation



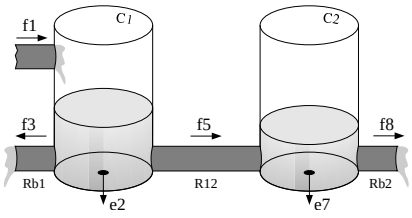
Intermezzo: Python - One page Summary

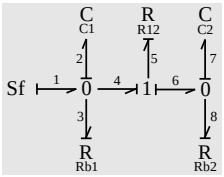
- Python Language Features:
  - Byte-Code Compiled/Interpreted
  - Interactive
  - Object Oriented
  - Dynamic Semantics
  - Supports Modules and Packages
- Rapid Application Development support through:
  - High level data structures and operations Python source looks like "executable pseudo code"
  - Strong embedding/extending capabilities

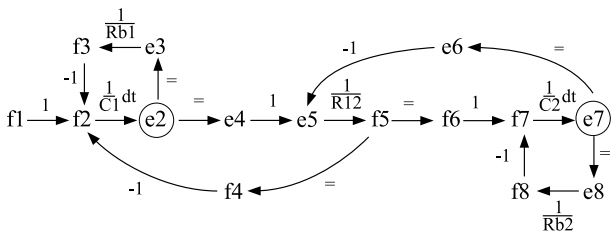
Actor with Embedded Python components

- How it works
  - Create Python class derived from `java.lang.object`
  - Compile to Java byte-code with **jpythonc**
  - Instantiate class as if native Java object
  - Application loads **jpython** package:
    Java implementation of the Python Virtual Machine
- Cool: Prototype Java components with RAD tool
- Catch: Performance hit over native Java

EXAMPLE: FDI of a multi-tank fluid system







EXAMPLE: FDI of a multi-tank fluid system (2)

Output of the Fault Isolation Process for fault $C2-$

```
> step 0
        f3:  0 .
        e7:  + .

> Active Hypotheses:
'C2-'
        f3: 0 + -  (0)
        e7: + - +  (0)
'C1-'
        f3: + - +  (0)
        e7: 0 + -  (0)
'Rb2+'
        f3: 0 0 +  (0)
        e7: 0 + -  (0)
'R12-'
        f3: 0 - +  (0)
        e7: 0 + -  (0)
'Rb1+'
        f3: - + -  (0)
        e7: 0 0 +  (0)
```

```
> step 1
        f3:  + +
        e7:  + - 1

> Active Hypotheses:
'C2-'
        f3: + + -  (0)
        e7: + - +  (0)
```


Summary and plans for further work

- Current implementation allowed us to make TRANSCEND suitable for stream based processing.
- Gaining insight in concurrency issues for Model Based FDI systems
- Future plans:
  - Integrate simulator for Bond Graph models (in progress):
  - Migrate towards discrete time dataflow
  - Construct online FDI system for the physical three-tank system testbed in our lab

Final Notes

Please visit the
Modeling and Analysis of Complex Systems group
online at:
http://macs.vuse.vanderbilt.edu

This project supported by:
Agilent Laboratories
NASA Intelligent Systems program
DARPA Software-Enabled Control program